IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re reissue application of

Group Art Unit: 2155

Charles F. Pyne

Application No. 09/512,967

Filed: February 24, 2000

For: **REMOTE FILE TRANSFER
METHOD AND APPARATUS**

Date: January 18, 2002

Examiner: David A. Wiley

37 C.F.R. § 1.131 DECLARATION OF CHARLES F. PYNE

TO THE ASSISTANT
COMMISSIONER FOR PATENTS:

I, Charles F. Pyne, declare as follows:

1. I am the inventor named in the above-identified patent application, which is a reissue of U.S. Patent No. 5,721,907. I submit this declaration to establish completion in the United States of the inventions claimed in this application before January 3, 1994, the effective date of U.S. Patent No. 6,076,084, for File Transfer Method and Apparatus Using Delimiters, cited by the Examiner in his October 23, 2001 Office action.

2. To establish that the date of conception of the invention disclosed in this application precedes January 3, 1994, I attach Exhibits A and B as evidence. Exhibits A and B were dated before January 3, 1994, but the dates have been intentionally obscured during the reproduction process. Traveling Software, Inc., to which references are made in Exhibits A and B, is a predecessor company to the assignee of this patent application. Traveling Software, Inc. employed me as an independent consultant to work on the project described in Exhibits A and B.

3. Exhibit A -- File Delta Algorithm Details includes three typewritten pages on which handwritten notes and drawings appear and one

page of hand-drawn block diagrams. Exhibit A is a design summary document that records information I had either written or conveyed orally to a Traveling Software, Inc. employee, who then prepared the document.

4. Exhibit A describes the steps and algorithms involved in one implementation of a method of copying a source file located at a first computer into a reference file located at a second computer. Page 1 sets forth a hand-drawn diagram showing the embodiment illustrated in Fig. 2 of this application and a description of the calculation of the Block Key Array in accordance with the Fig. 2 embodiment. Page 2 sets forth a hand-drawn diagram showing the embodiment illustrated in Fig. 3 of this application and a description of the generation of the sliding-window in accordance with the Fig. 3 embodiment. Page 3 sets forth two hand-drawn block diagrams. The block diagram on the top half of page 3 shows the embodiment illustrated in Figs. 4 and 5 of this application and describes an exemplary routine and subroutine for implementing the file transfer program in accordance with the Figs. 4 and 5 embodiment. The block diagram on the bottom half of page 3 shows the embodiment illustrated in Fig. 6 of this application and describes an exemplary routine for determining the differences in the source and reference files and transferring those differences to the receiving computer where a destination file is created in accordance with the Fig. 6 embodiment.

5. Exhibit B -- Letter dated before January 3, 1994 from Megan Cate of Traveling Software, Inc., to its then patent attorney, Bruce O'Connor, requesting that a patentability search be performed on the File Data Concept implemented with the File Data Algorithm. The letter sets forth a brief description of the subject matter of Exhibit A. I had prepared for use by Traveling Software, Inc. the "Background," "The File Delta Concept," and "The File Data Algorithm" portions set forth in the Letter. Exhibit B generally describes (1) the technological state of the art of file data transfer when the Letter was written and (2) a method of identifying and isolating the differences in content between two files and of transmitting only the differences in content from a first computer to a second computer.

6. The inventions disclosed and claimed in this patent application were conceived and first reduced to practice in the United States before January 3, 1994. This assertion is confirmed by the first sentence in the fifth paragraph on page 4 of Exhibit A, which states: "In [obscured date] the File Delta algorithm was implemented as part of a working demonstration prototype running under Microsoft Windows." The obscured date is prior to January 3, 1994.

7. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: January 19, 2002

Charles F. Pyne
Charles F. Pyne

Exhibit A, page 1

File Delta Algorithm Details

[redacted]

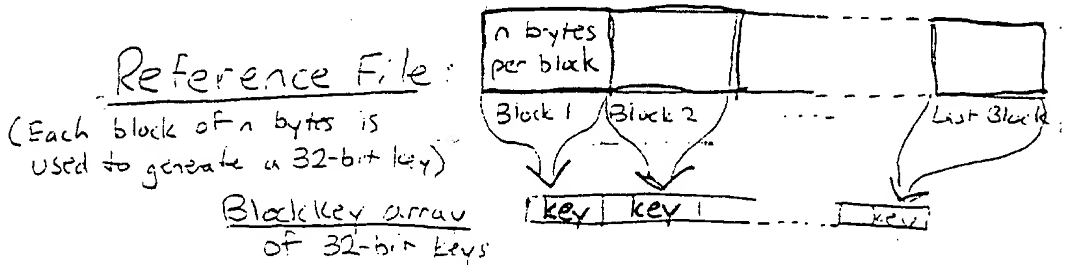
Description

This document provides more details on the file delta technology outlined in the document by Charlie Pyne. The assumption is made that the reader has already read this document.

The Algorithm in Detail...

Below is outlined the sequence of steps for the copy delta algorithm involved in copying a file from *computer A* (which contains the newly modified *source file*) to *computer B* (which contains the original *reference file*).

COMPUTER A	COMPUTER B
	(1) Compute Keys The *BlockKey Array* is created consisting of a 32-bit *key* value for every *block* (ie *n bytes of data, where n is some constant such as 256)* in the *reference file.*


Reference File:
(Each block of n bytes is used to generate a 32-bit key)
n bytes per block
Block 1
Block 2
....
Last Block
BlockKey array of 32-bit keys
key
key
....
key

COMPUTER A	COMPUTER B

	The key is calculated by simply adding the value of each byte as it is encountered; thus, each block starts with a key of 0, and each byte value in the block is added to the key in sequence like:
	KEY = 0
	WHILE NOT END OF FILE AND NOT END OF BLOCK
	KEY = KEY + NEXT_BYTE_FROM_FILE;
	END.
	
	Once the *BlockKey Array* has been created using the entire *reference file*, it is sent to computer A.

Receive BlockKeys <<<	<<< (2)transfer BlockKeys to computer A

Exhibit A, page 2

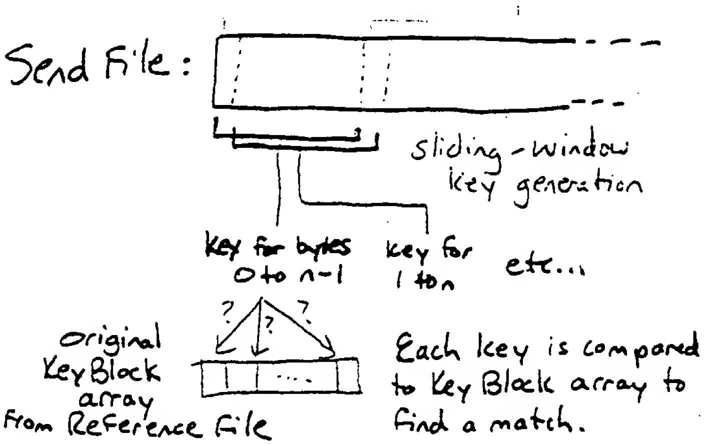
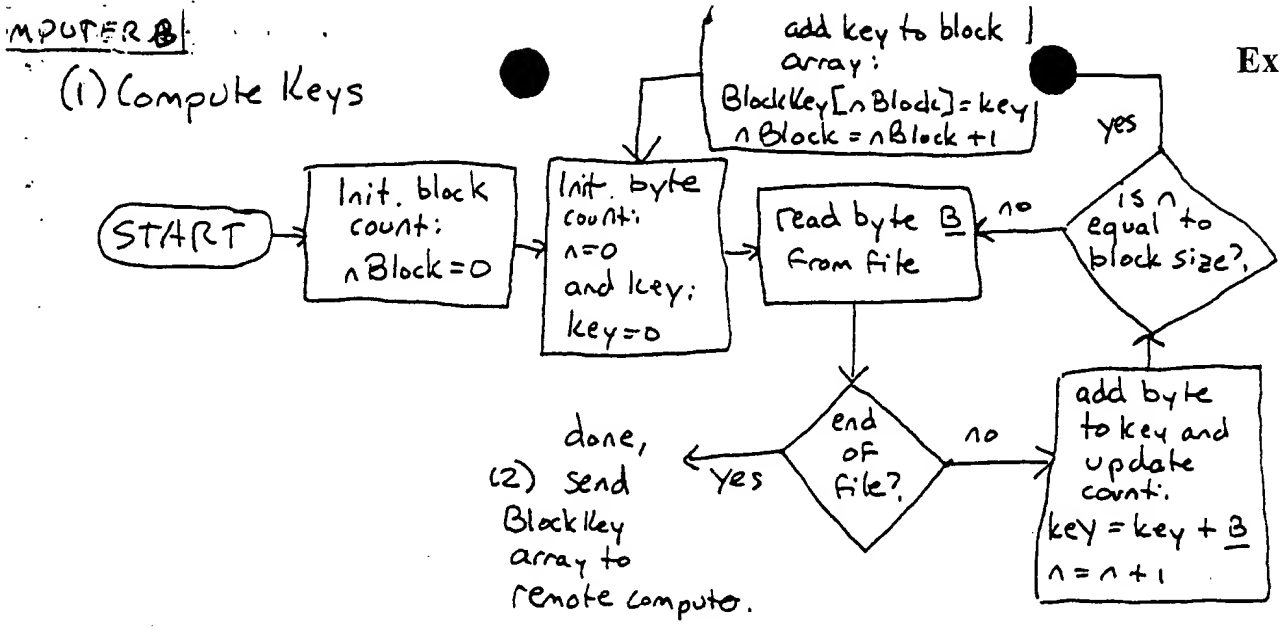
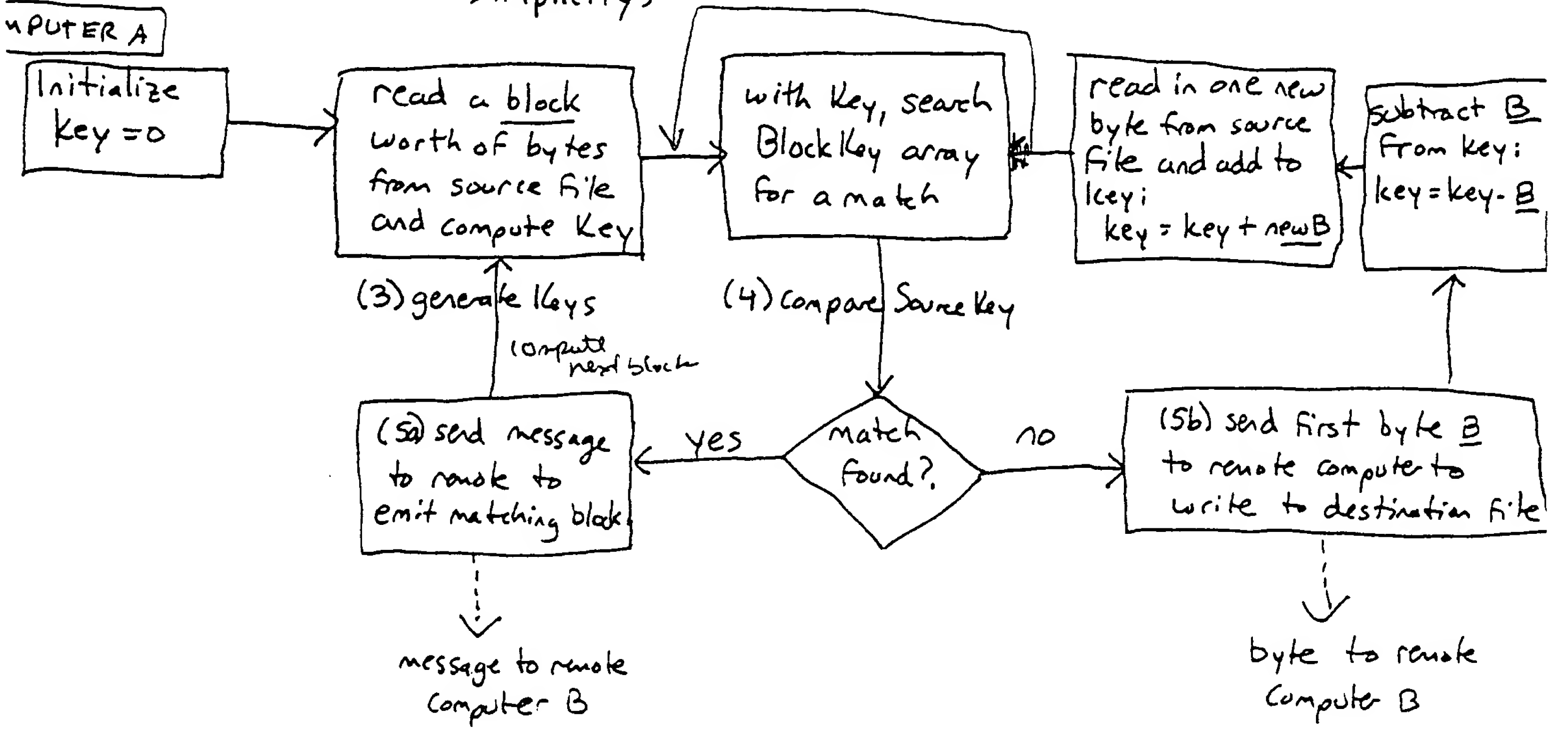
COMPUTER A	COMPUTER B
<u>(3) Generate Keys for source file..</u> With the *BlockKey Array* on computer A, data is read from the *source file* and a key is generated for the first *block (n bytes)* of the file, using the exact same key algorithm that was used to generate the original *Block Key Array* (ie each byte is added to the key value). <u>(4) Compare Source Key to KeyBlock Array</u> Once the *key* for the first *block* has been calculated, this *source file key* is compared with every key in the *Block Key Array*. If a match is found, then a message is sent to *computer B* that a block in the reference file has been matched, and we can start calculating a new *source file block key* for the next *block (n bytes)* of data...	 Send File: sliding-window key generation Key for bytes 0 to n-1 key for 1 to n etc... ? ? ? original KeyBlock array From Reference File Each key is compared to Key Block array to find a match.
<u>(5a)Send matched block message >>></u>	<u>>>> Receive matched block message</u> A *block* on the *source file* has matched a *block* in the *reference file* and can be written to the *destination file.*
...if the *source file key* does <u>not</u> match a key in the *Block Key Array*, then the *first* byte from the beginning of the *block* is *sent* to *computer B* to be written. The value of this byte is subtracted from the *source file key* and a new byte is read from the *source file* and added to the *source file key*. This is how the ***sliding window*** feature is implemented. We now have a *source file key* for the next block (shifted one byte over), and we repeat the comparison of keys above (4). If no key match is found then the byte is sent to the remote computer and the *block* window is shifted one byte again... <u>(5b)Send unmatched byte >>></u>	<u>>>> Receive unmatched byte</u> Receive unmatched byte and write it to the *destination file*

Exhibit A, page 3


MPUTER B
(1) Compute Keys
START
Init. block count: nBlock=0
Init. byte count: n=0 and key: key=0
read byte B from file
end of file?
yes
done, (2) send BlockKey array to remote compute.
no
add byte to key and update count: key = key + B n = n + 1
is n equal to block size?
no
yes
add key to block array: BlockKey[nBlock] = key nBlock = nBlock + 1

(Note: key computation details and end-of-file handling left out of details below for simplicity)


MPUTER A
Initialize key = 0
read a block worth of bytes from source file and compute key
(3) generate keys
compute next block
with key, search BlockKey array for a match
(4) compare Source key
match found?
yes
(5a) send message to remote to emit matching block
message to remote Computer B
no
(5b) send first byte B to remote computer to write to destination file
byte to remote Computer B
subtract B from key: key = key - B
read in one new byte from source file and add to key: key = key + newB

FILE DELTA FLOW DIAGRAMS

[redacted]

Traveling Software, Inc. CONFIDENTIAL

Exhibit A, page 4

File Keying: The use of keys to represent the contents of the reference file reduces the amount of data sent on the reverse channel by a factor of about 50:1.

Sliding Window Matching: The File Delta Tx Filter employs a sliding window matching technique to compare keys for the source file with keys from the reference file. Using this technique the algorithm can identify matching sections of the two files that are in different positions. This means that the two files can be successfully compared even when insertions and/or deletions have taken place.

Incremental Key Computation: Since key or checksum computation can be a time consuming task, the File Delta algorithm uses an incremental key computation method. This reduces the Tx Filter computation time by a factor of more than 200:1.

Development History

The File Delta technology was conceived by Charles Pyne in [redacted]. The algorithm was defined and refined over the next year. The technology is the result of original work and no reference was made to outside sources.

In [redacted] the File Delta algorithm was implemented as part of a working demonstration prototype running under Microsoft Windows. The prototype code was written for the purpose of illustrating the key concepts and performance characteristics of the algorithm.

Exhibit B, page 1

TRAVELING SOFTWARE

The Leader in Portable Communications

[redacted]

Bruce O'Connor
Christenson, O'Connor, Johnson & Kindness
2800 Pacific First Centre
1420 Fifth Avenue
Seattle, WA 98101-2347

Re: File Delta patent search

Dear Bruce:

TSI consulting engineer Charles Pyne invented a concept to enhance the transfer of data between files over a modem. This concept is described below. We request that you perform a patentability search for the concept. I would appreciate receiving an estimated deadline for the search completion. We want this information as soon as possible. If there is an extra fee for expediting the search, let me know so I can obtain the necessary pre-approval.

Background

Many communications software products provide the facility for transferring files between remote computers by means of modems connected to telephone lines. Since modem transmission rates are relatively slow, the time required to send a file can be significant. For this reason products often incorporate some form of data compression to reduce the amount of data sent and thus the time required for transmission. Traditional data compression algorithms take advantage of the internal redundancy of a file and produce a compressed representation of the original. Typical compression ratios are on the order of 2:1.

The File Delta Concept

Practical applications of remote file transfer often involve a situation where the receiving computer already contains a file that is very similar (or identical) to the file being transmitted. For example, the transmitted file may be a revision of a text file with a few words or sentences changed. The File Delta concept is to identify and isolate the differences between the two files and to transmit only the changes. For similar files this can result in compression ratios far in excess of those achieved by traditional data compression methods.

Exhibit B, page 2

The File Delta Algorithm

The File Delta Algorithm can be incorporated as a module in a file transfer program such as LapLink. The main program designates a file to transmit along with a file on the remote computer that is purportedly similar. The File Delta Algorithm employs a *remote differencing detection* technique to identify similarities and differences between the two files. It then transmits a *delta data stream* to the remote computer. On the receiving computer, the algorithm merges the delta data stream with its version of the file to produce an exact copy of the transmitting computer's version of the file.

The algorithm can detect changes to parts of a file as well as insertions and deletions. The only requirement is that the two files contain data sequences of a few hundred characters in length that match each other. The algorithm is more effective with text files. Testing with typical cases has shown that compression ratios of 5:1 are common and ratios of up to 50:1 are possible for very similar files. In cases where the two files are not sufficiently similar there may be an expansion of a few percent.

Please call me at 487-5414 if you have any questions.

Sincerely,

Megan Cate

Megan Cate
OEM Sales & Contract Administrator

cc: Dave Payne
Mark Eppley
Raju Gulabani